REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	3.Nov.99	THESIS

4. TITLE AND SUBTITLE
GROUND BASED INTERCEPT OF A BALLISTIC MISSILE: SIMULATION TRUTH/MODEL INTERFACE

5. FUNDING NUMBERS

6. AUTHOR(S)
2D LT CONE KYLE M

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
UNIVERSITY OF COLORADO AT COLORADO SPRINGS

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
THE DEPARTMENT OF THE AIR FORCE
AFIT/CIA, BLDG 125
2950 P STREET
WPAFB OH 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
FY99-391

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
Unlimited distribution
In Accordance With AFI 35-205/AFIT Sup 1

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

19991117 099

14. SUBJECT TERMS	15. NUMBER OF PAGES
	22
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT

Creative Investigation Abstract

Cone, Kyle Matthew (M.E., Engineering Space Operations)

Ground Based Intercept Simulation, Simulation Truth/Model Interface

Thesis directed by Doctor Don Coughlin

As the simulation truth/model interface architect, I was charged with generating truth data for an intercontinental ballistic missile in flight. Further, I was responsible for presenting the Ground Based Interceptor simulation in a visual format. I used the *Satellite Tool Kit* software to present the results of the simulation in a detailed, graphical format, and I used the supplementary *Missile Flight Tool* module to model an intercontinental ballistic missile and generate its corresponding truth data.

GROUND BASED INTERCEPT OF A BALLISTIC MISSILE:

SIMULATION TRUTH/MODEL INTERFACE

by

KYLE MATTHEW CONE, 2LT, USAF

B.S., Purdue University, 1998

A thesis submitted to the Graduate Faculty of the

University of Colorado at Colorado Springs

in partial fulfillment of the

requirements for the degree of

Master of Engineering in Space Operations

Department of Mechanical and Aerospace Engineering

1999

Cone, Kyle Matthew (M.E., Engineering Space Operations)

Ground Based Intercept of a Ballistic Missile:
Simulation Truth/Model Interface

Thesis directed by Doctor Don Coughlin

As the simulation truth/model interface architect, I was charged with generating truth data for an intercontinental ballistic missile in flight. Further, I was responsible for presenting the Ground Based Interceptor simulation in a visual format. I used the *Satellite Tool Kit* software to present the results of the simulation in a detailed, graphical format, and I used the supplementary *Missile Flight Tool* module to model an intercontinental ballistic missile and generate its corresponding truth data.

CONTENTS

INTRODUCTION 1

DEFINITION OF MY ROLE 2

SIMULATION TRUTH 3

BACKGROUND 4

Satellite Tool Kit 4

Missile Flight Tool 5

PROCESS 8

Overall Integration Framework 8

Interfaces 14

Outputs from Simulink/Stateflow 14

Modeling of Radars 16

Modeling of IR Sats and Sensors 19

Modeling of EKV 21

Modeling of GPS Sats and Battle Manager 21

ANALYSIS AND CONCLUSIONS 22

Troubles Encountered During Mission 22

Growth Possibilities 23

FIGURES

FIGURE

1. Sample of Poost-Boost Phase Sequence 11

2. Reentry vehicles descending to their targets 13

3. Radars have access to ICBM and actively track it 18

4. The search radar (orange) and track radar (white) track the enemy ICBM 19

Introduction

The Ground Based Intercept (GBI) simulation was a team-effort simulation that asked its simulators to address the technical issues associated with the detection, acquisition and hit of an incoming ballistic missile. Also, the Ground Based Intercept simulation satisfied the creative investigation requirements portion of the Master of Engineering Space Operations degree. Each GBI team member was given a specific role and a specific area of interest to examine. The team members and their respective roles and areas of interest are defined below:

1. Program Manager - Captain Mark Wert
2. System Engineer - Captain Tim Fromm
3. Simulation Truth/Model Interface Architect - Lieutenant Kyle Cone
4. Control Engineer (vehicle control) - Lieutenant Scott Klempner
5. Radar Engineer (Tracking, Discrimination) - Lieutenant Brian Egbert
6. IR Engineer (Detection, Tracking) - Lieutenant Dan DeYoung
7. Battle Manager (Sensor Fusion) - Lieutenant Michelle Roxburgh
8. GPS Engineer - Surachai Sukchoo

This paper only details the Simulation Truth/Model Interface portion of the GBI creative investigation. The

remaining portions of the investigations are not included in this document; however, one can obtain information on any portion of the GBI simulation simply by contacting each portion's manager.

Definition of my role

As the Simulation Truth/Model Interface architect, I was charged with providing the different simulation components with a visual forum to display and view their roles in the simulation. These different simulation components include infrared satellites and sensors, the battle manager facility, the search and track radars, the GPS satellites and the Exo-atmospheric Kill Vehicle (EKV). To accomplish the task of providing a visual forum, I used the *Satellite Tool Kit* (STK) satellite systems analysis software and the supplementary *Missile Flight Tool* (*MFT*) module provided by Analytical Graphics, Inc (AGI).

Furthermore, I was tasked with providing the position and velocity data of the enemy Intercontinental Ballistic Missile (ICBM) to the IR satellites and the search and track radars in the simulation. Passing this data to each was accomplished by modeling the flight of the ICBM in *MFT*, passing that data to *STK*, saving the data as a text file, modifying it to a *MATLAB* m-file and, finally, passing it into a *MATLAB Simulink* model.

It is here, in *Simulink*, that the data was used to continue the simulation.

Simulation truth

In order to begin the simulation, it was necessary to have true data that accurately portrayed the position and velocity of the incoming ICBM. This data was simply the ephemeris from the ICBM and its reentry vehicles (RVs) as it traversed its trajectory as modeled in *MFT*. Specifically, the truth data included the time, latitude, longitude, altitude, latitude rate, longitude rate and altitude rate of the ICBM and both the actual and decoy RV. Our entire Ground Based Interceptor (GBI) simulation relied on the truth data for an accurate depiction of the ICBM's flight and both RV's deployment and descent.

A master truth file, containing specific missile-status information could have been created. This master truth file would have contained data on each of the ICBM's components - each of the three stages, the shroud and each of the RVs. Moreover, this master truth file would have data on each of the missile's part's ascent, separation and descent. Therefore, with this master file, one would have known the entire history of the ICBM. One would have known where it was at each time step and what its status was, including stage separation, shroud separation and RV status. However, I obtained only data that

was pertinent to our simulation, and that resulted in two truth files, one for the primary target RV and the other for the decoy RV.

Background

Satellite Tool Kit

In order to see the ICBM launch, fly and separate, and in order to see the target RV and the decoy RV deploy, a program capable of presenting results in a detailed, graphical format was necessary. Furthermore, this program needed the ability to propagate satellites, model sensors in different environments and display, on-screen, when a sensor could "see" the ICBM. The *Satellite Tool Kit* software went above and beyond the aforementioned criteria; therefore, the *STK* software was chosen to visualize the actions of the different simulation components during the simulation.

Satellite Tool Kit is a commercial-off-the-shelf satellite systems analysis tool used by the space industry to visualize, model, simulate and analyze complex scenarios involving an extensive range of options above and beyond the obvious satellite. The *STK* software allows a user to not only propagate a satellite's position in time, but also determine the time one object can see or "access" another object; this tool became extremely useful in our GBI simulation. Further, *STK* allows

modeling of other vehicles and objects such as airplanes, ships, ground vehicles, facilities, planets, stars, receivers, transmitters and sensors.

STK provides a user with many options, one of which is to model different behaviors. These behaviors include, but are not limited to, drag affects on an object, solar radiation affects on a satellite and any third-body gravitational affects. A user has the option to define the coordinate system in which he/she would like to orient his/her object as well. Not only can a user add affects to his/her scenario, but one can also set constraints on objects. These constraints include, but are not limited to, positions of the Sun and Moon, time-based constraints on a facility or target and standard constraints such as minimum and maximum angles and altitudes.

Missile Flight Tool

In order to launch and propagate the ICBM based on actual flight profiles, a program that modeled "real world flight" of an actual ICBM was critical. The ability to model the deployment and descent of reentry vehicles was also critical to the task of modeling an actual ICBM. To have *STK* visualize the launch and flight of the ICBM, a program that was already a part of *STK*, or one that could interface with *STK* was necessary, too.

The *Missile Flight Tool* module of *Satellite Tool Kit* satisfied our needs.

"*Missile Flight Tool* is a high-fidelity missile flight path generator…"[1] that enables the user to easily understand complex missile operations. By integrating *MFT* with *STK*, a set of unclassified databases is opened to the user. These databases represent a wide range of different missile types and performance capabilities that allows one the capacity to generate multiple-stage missile trajectories. Further, a user can easily analyze and visualize complex relationships between any portion of the missile phase, operations and satellite systems with the integration of *MFT* and *STK*.

MFT is an easy tool to use for the established and experienced user. It is the installation of and initial use of *Missile Flight Tool* that can cause the user a great deal of anxiety and stress. Since *MFT* is an *STK* module, the use of *MFT* requires a password and either a computer station host ID number or an expiration date. In fact, *Missile Flight Tool* is so sophisticated that it requires a special export license agreement with AGI. Certain aspects of the *MFT* module and details of the missile databases contained within the module are considered 'sensitive material.' Therefore, only certain countries and persons are cleared to use this module.

Unlike the *Satellite Tool Kit* ballistic missile propagator, which simply flies a vehicle on an elliptical path beginning and ending at the Earth's surface, the *Missile Flight Tool* missile propagator models ICBMs properly, meaning, it models the "real world flight" of an ICBM. To illustrate, the *MFT* module offers staging phases and reentry vehicle capabilities while the standard *STK* ballistic propagator version does not. Moreover, the supplementary module is able to model eleven different missiles. Each missile is defined by its maximum range, number of stages, number of RVs and guidance type. The missiles vary between short test missiles with a maximum range of 300 kilometers and strategic ICBM missiles with a maximum range of 12000 kilometers.

MFT provides a user with many other options, one of which is to model different behaviors and forces acting on the missile in question. These behaviors and forces include, but are not limited to, an oblate versus a spherical Earth and atmospheric density, pressure and temperature. Also, *MFT* uses a wide variety of functions and procedures that deal with the physical shape and characteristics of the Earth.

Process

Overall integration framework

The Ground Based Interceptor simulation began and will end using *Satellite Tool Kit*. It was here, in *STK*, or *MFT* rather, that I generated the truth data. This truth data, along with system time, was then input into *Simulink* to continue the simulation. In the end, the different simulation components will output each of their data to text files, and I will, then, input this data into *STK*. The end result will be a complete, dynamic display of our GBI simulation, including a comparison of the different truth data versus actual data; the incorporation of GPS truth data will be described later.

The GBI simulation began with opening the *MFT* and *STK* applications. Initially, many others and I thought that the *Missile Flight Tool* module was already installed on the Master of Engineering Program Office's computer stations. Therefore, I obtained the necessary *MFT* password, and tried to open the module; this did not work; *MFT* was not on the computer stations. Dr. Charles Fosha, director of the Master of Engineering program, helped me request and receive the necessary *MFT* CD from Analytical Graphics, Inc. Simply following the installation procedures did not work, though. To explain, the computer station I installed the module on did not recognize that I had installed the module on it! To alleviate this problem, I had to

call AGI technical support to only find out that I had to manually install the password myself. To ensure this situation did not happen again, I wrote down the "new" installation instructions for *MFT*. A five-minute installation job took over three weeks to accomplish.

With both applications open, *Satellite Tool Kit* and *Missile Flight Tool* interface with each other through the *STK Connect* module. This module allows a simple IPC connection. What is not so simple is establishing this connection. To the established user, the connection is made with two clicks of the mouse. For the non-established user, one must first obtain the password and password expiration date from Analytical Graphics, Inc. Once I received this password, I tried establishing the connection; however, I ran into the same problems here as I did with installing *MFT*. As a result, I had to manually install the password and expiration date for this module, as well.

Once I had *MFT* installed and the *STK/MFT* interface established, I began my quest for truth data by inputting our launch and impact sites. We chose to simulate an ICBM launch from Paris, France; New York City, USA was chosen to be the primary impact site, or target, and Washington D.C. was chosen as our decoy impact site. We wanted to simulate an intercontinental ballistic missile strike, but there was not any further rhyme or reason as to why Paris should strike New York.

Nor was there any reason as to why we chose Washington D.C. as our decoy impact site. My first choice for launch and impact points were New York City and Tokyo, Japan, respectively. However, for politically correct reasons, our GBI team rejected the idea, and I discarded the New York/Tokyo scenario.

Originally, our ASE 583 Engineering Simulation professor, Dr. Don Coughlin, hinted that we might want to launch our enemy missile from Hawaii and have it target New York City. During our Critical Design Review (CDR) presentation, Dr. Coughlin noted that doing so would have made the intercept portion of our GBI simulation easier. I failed to recognize this and pressed forward in setting up the initial parameters for our simulation.

Once I chose the launch and impact sites, I had to determine the type of missile that would be modeled. I chose to model the LGM-30 Minuteman III intercontinental ballistic missile as our enemy ICBM because it closely resembled an *MFT* missile option. Specifically, I selected the *MFT* LRM_2-12000 missile model. This missile "…is a strategic ICBM-type missile with a 12000 km maximum range, three stages and a PBV [Post Boost Vehicle] with three RVs."[1] The actual Minuteman III missile has a range of 10000-plus kilometers, three stages and is capable of carrying three warheads.[2]

Our simulation called for us to model only two RVs, one primary, one decoy. Since the LRM_2-12000 missile carried three RVs, I had to modify the ICBM model in *MFT*. I targeted two of the warheads to impact New York City (one became redundant),

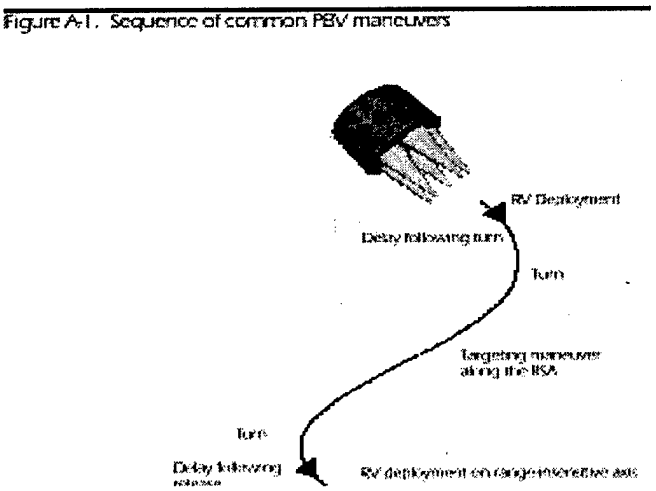

Figure 1. Sample Post-Boost Vehicle sequence

while the third satisfied the decoy requirement in our simulation. *MFT* is able to model four types of Post-Boost Vehicle configurations, meaning, there are four different ways the reentry vehicles can be deployed. Figure 1 (reprinted from Appendix A of the *MFT* user's manual) illustrates a possible Post-Boost Vehicle sequence. To ensure the viewer attains an accurate representation of the RVs' deployment and descent, I instructed *STK* to display only the decoy and one of the primary RVs.

I, initially, wanted to model the flight of the LGM-118A Peacekeeper ICBM. What attracted me to this missile was its ability to deliver up to 10 RVs with greater accuracy than any

other ballistic missile. It, too, has three stages, and like the Minuteman III, has a range of 10000-plus kilometers. With the end of the Cold War, however, the United States agreed to eliminate the Peacekeeper missiles and institute the Minuteman as the only land-based ICBM in the nuclear Triad. The desire to keep the simulation as true-to-life as possible drove me to reject the Peacekeeper.[3]

The final 'initial' decision our GBI team had to make was which size time step we wished to propagate our simulation with. We determined that the EKV needed a relatively small time step to accurately achieve a close-encounter approach with the enemy ICBM. Therefore, we selected a time step of 0.1 seconds. Unfortunately, I was unable to get *Missile Flight Tool* to model the ICBM's flight in such a small time step. In light of this, our time step became the time step that *MFT* was able to propagate with, namely, one second.

The above decisions enabled me to propagate the ICBM and attain the truth data needed for the rest of the simulation. I ran into a lot of trouble in propagating the ICBM, though; I learned that *MFT* is very sensitive. One must follow a certain process of inputting data before *MFT* will fully compute the trajectory of the ICBM. Also, I found many nuances of the *Satellite Tool Kit* operating system during the link-up between *STK* and *MFT;* some of these nuances are detailed later in this

paper. After many hours of frustration, I arrived at the solution to my *STK/MFT* operating dilemma; I did the smart thing of writing down, step-by-step, the correct way to work with *MFT* and *STK*. With a click of a few buttons, I obtained the truth data and exported it to *Satellite Tool Kit*. Figure 2 displays the end result of propagating a missile and visualizing it using *STK*.

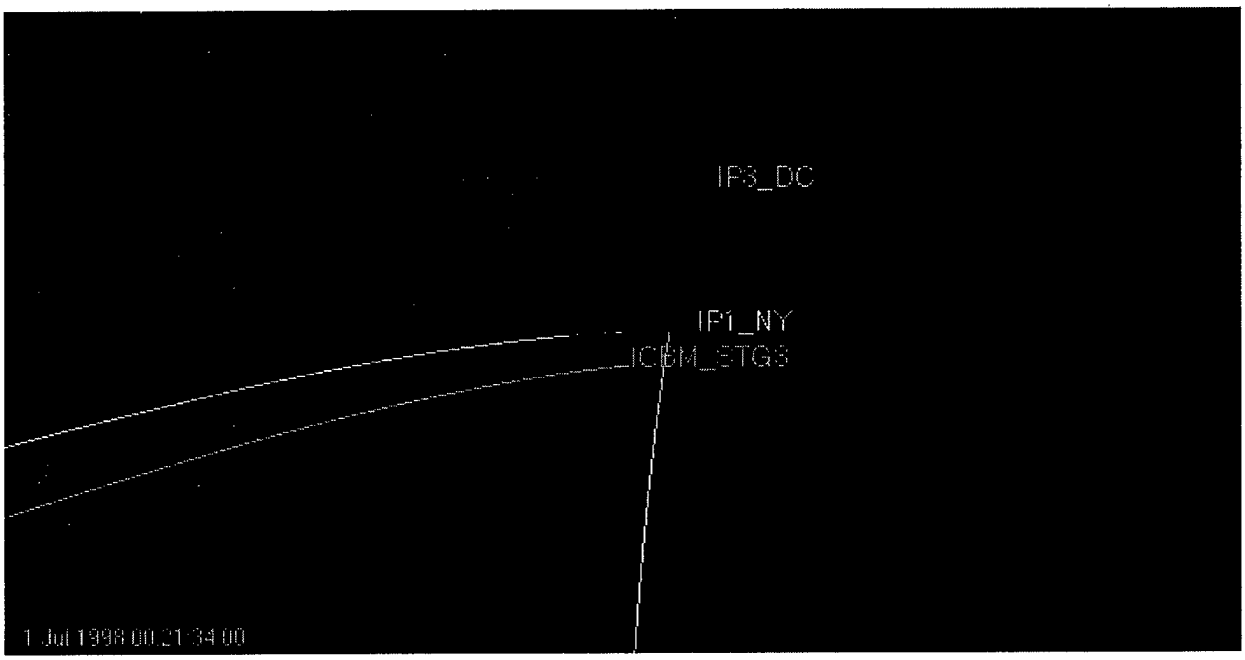

Figure 2. Reentry vehicles descending to their targets.

RV_3 is the decoy reentry vehicle on its way to Washington D.C., while RV_1 is the primary target RV. One can see in Figure 2 that the third stage descends after it has deployed the RVs.

Interfaces

We used the *MATLAB Simulink* and *Stateflow* programs and the ICBM truth file to arrive at each component's output. Unfortunately, the versions of *STK* and *MATLAB* that were available to our GBI team did not directly interface with each other. Therefore, before *Simulink* could read the truth data, I had to transform the data into a *MATLAB* m-file. This was a simple process of first saving the truth data as a text file, then, modifying that text file as *MATLAB* code. Presently, a *MATLAB-STK* interface is on the market; we were unable to get a hold of it because of its staggering $80000 price tag.

I helped others create their models in the *Simulink* program once *Simulink* was able to read the truth file I had just created. I was able to provide my GBI team members with advice on how best to write *MATLAB* code, and Lieutenant Scott Klempner and I, with the help of Dr. Coughlin, were able to figure out how to get *Simulink* to run and evaluate a *MATLAB* m-function file.

Outputs from Simulink/Stateflow

To output the data generated by the different simulation components in *Simulink* and *Stateflow* to *STK*, each data file was saved as a text file. Each file had to be in a particular *STK* readable format, though. For example, the GPS satellites in our

simulation were not created in *STK*. Instead, their locations and orbits were defined in *MATLAB*. To display their locations at each time step in *STK*, each GPS satellite's ephemeris must be imported to the *STK* application; importing data is done by creating a text file with the necessary information in the necessary location. In our case, the satellites' latitude, longitude, altitude, latitude rate, longitude rate and altitude rate is needed at each time step. The exact format for an ephemeris file (.e) can be found on pages C-9 to C-17 in the *STK User's Manual*.[4]

Further, to model the infrared satellites (IR sats) correctly, the azimuth and elevation at each time step of the onboard sensor's cone angle must be imported into an *STK* sensor. The azimuth and elevation will be generated in *Simulink* and saved in an *STK* readable format, namely, a text file.

The following items will be imported into *STK* once the *Simulink* output files are generated: the EKV and its flight path, the pointing angles of the IR sensors and search and track radars and the GPS satellites. The EKV data file that will be imported into *STK* will be formatted similar to the ICBM truth file. The EKV data file will contain the kill vehicle's latitude, longitude, altitude, latitude rate, longitude rate and altitude rate at each time step. By importing the pointing angles of the IR sensors and the search and track radars, we

will able to accurately display when a sensor or radar is able to see the ICBM. Knowing when the sensors and radars have access to the ICBM aids in the ease of understanding our GBI simulation.

We had to decide on locales for each of the sites - the radar site, the locations for the IR sats and the EKV site - before we could begin to think about importing any data into *STK*. Our decisions were based on a combination of technical and non-technical reasons. Access times with the ICBM drove the location of the radar site; resolution, pointing angles and access times drove the placement of the IR sats and kill time drove the location of the EKV.

Modeling of radars

Lieutenant Brian Egbert modeled the radars located at Daqortoq, Greenland as phased-array radars. Unfortunately, *STK* does not have the option to model a phased-array radar. To compensate, I decided model the radars as sensors until such a time when actual radar data from *Simulink* is available for import. The sensor options available in *STK* are:

1. Complex Conic
2. Rectangular
3. Half-Power
4. Synthetic Aperture Radar
5. Simple Conic

I chose to model the search and track radars as simple conic sensors since this was the easiest option to understand and work with.

Before modeling the radars in this fashion, I ensured that the viewer would still able to able see and understand, visually, how the search and track radars play their part in our simulation. Lieutenant Egbert and I decided to place our radar installation at Daqortoq, Greenland after initially placing the radar installation at Thule, Greenland. We defined the parameters for the two sensors, one search and one track, and analyzed the times the two sensors had access to the ICBM. Placing the site at Thule did not allow us quick enough access, and the sensors could only see the enemy ICBM for a short period of time.

Figure 3, below, is a two-dimensional depiction of the search and track radars seeing the ICBM and tracking it along its trajectory.

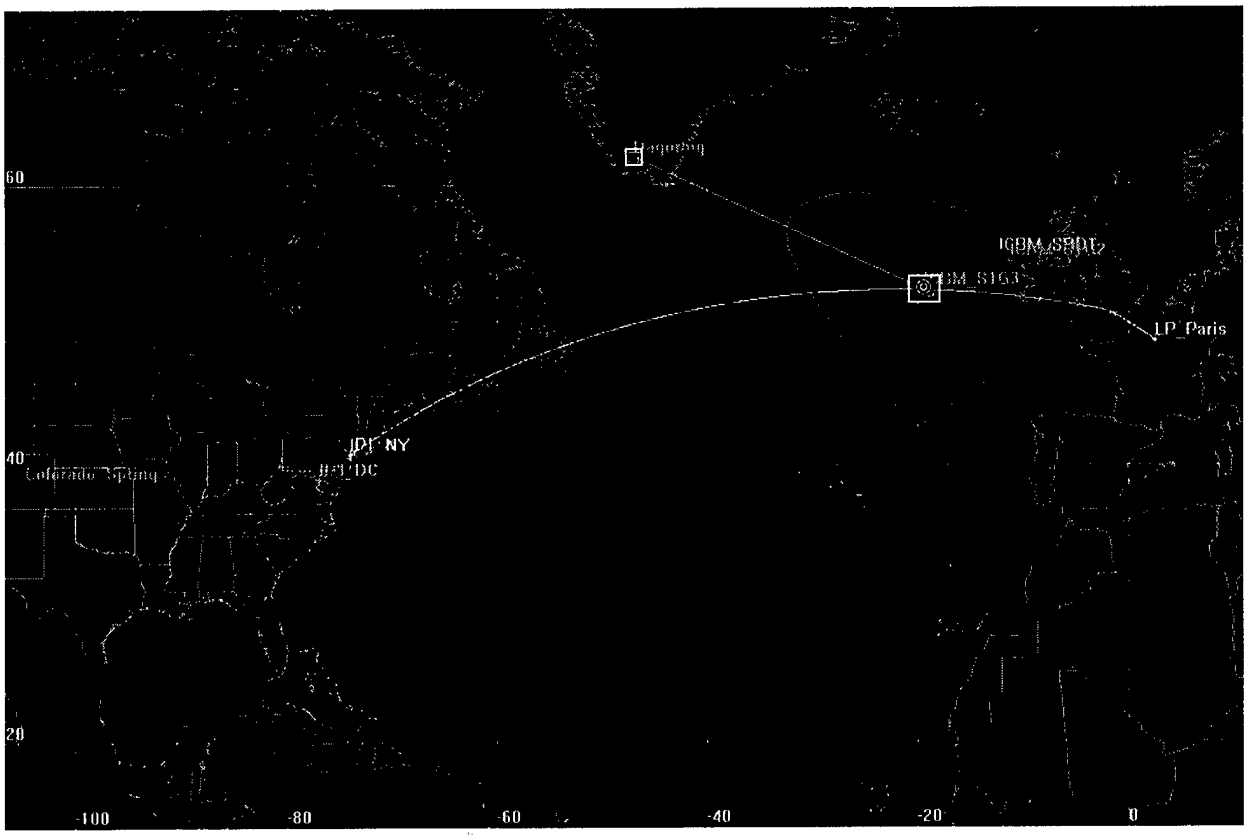

Figure 3. Radars have access to ICBM and actively track it.

The orange parabaloid around the missile shows that the search radar can see the ICBM. The white box around both the Daqortoq site and the missile shows that the track radar has access to the ICBM. The white line in between the two tells the viewer that the Daqortoq site is actively tracking the ICBM. As a point of interest, one can see in Figure 8-3 as well, the second stage and the shroud of the enemy ICBM have already separated from the missile.

Figure 4, below, is a three-dimensional version of the scene depicted above. In this figure, however, the viewer can

readily see, not only the trajectory of the ICBM, but the orange cone angle of 20 degrees that represents the search radar and the 1 degree cone angle that represents the track radar.

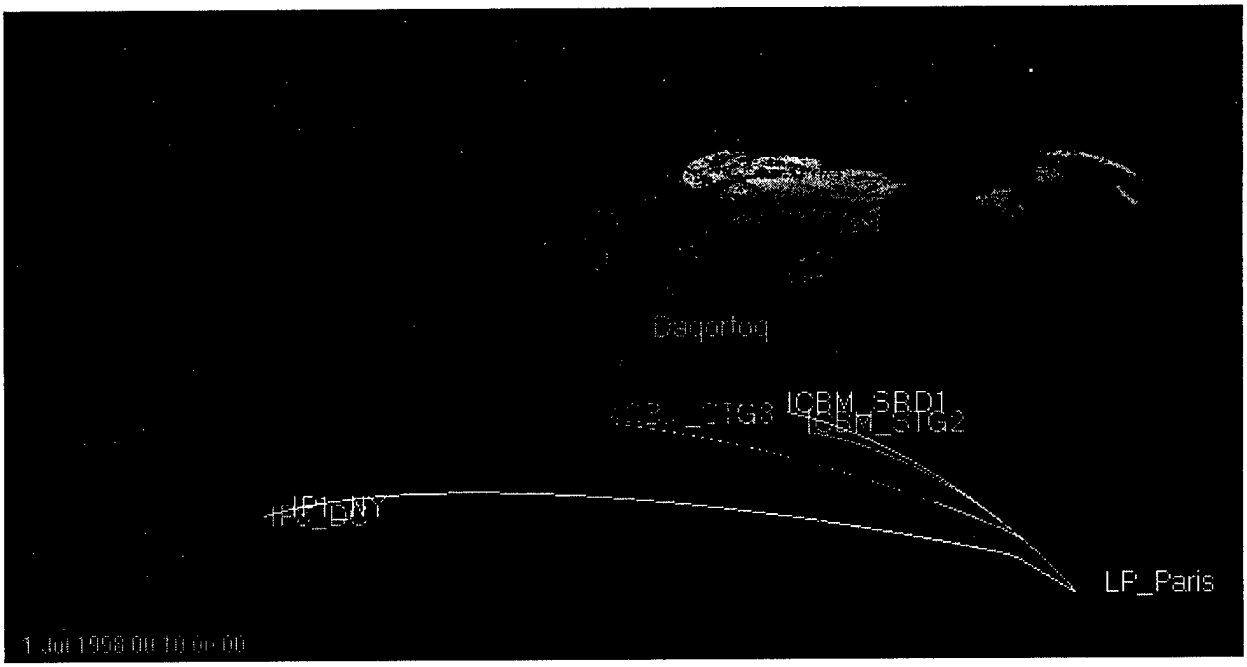

Figure 4. The search radar (orange) and track radar (white) track the enemy ICBM.

Again, as a point of interest, one can readily see the trajectory of the different stages of the ICBM. The white line spanning the Atlantic Ocean is the ICBM's ground track.

Modeling of IR sats and sensors

The infrared sensors designed to detect the launch of the enemy ICBM were placed onboard geostationary satellites. Lieutenant Daniel DeYoung and I saw that, from this altitude, the IR sensors could trace the entire surface of Earth. *STK's* different sensor options allow me to somewhat model the

spiraling action of the actual IR sensor until I receive actual azimuth and elevation pointing angles of the sensor from an *Simulink* output file. With the options given me by *STK*, I modeled the spinning movement of the IR sensor about its boresight by defining a spin rate and a single set of pointing angles.

Working together, the two sensors see the ICBM throughout its flight. One of the downfalls of *STK*, however was that once the sensors' spin rates and pointing angles were defined, I could not model the actual 'workings' of the sensors. To explain, once I defined the initial spin parameters for the IR satellites' sensors, I could not model the transition from searching to tracking the missile with the IR sensors; that is, I could not model the IR sensors attaining a 'lock' on the target. This problem will be relieved when I receive the IR pointing angles from the *Simulink* output file. With this data I will be able to model the proper spiral spin of the sensor and model its transition from a search mode to a tracking mode.

This is not to say that I can not model the IR sensors attaining a lock on the target without the *Simulink* data. Adding two more satellites, each with its own sensor, to the scenario will relieve this problem. All I have to do is define these additional sensors with the proper parameters so that they "turn on" once they see the ICBM. Keeping in mind our goal of

simulating a real world event, I choose not to travel that route.

Modeling of EKV

The Exoatmospheic kill vehicle launched from New York City once it determined it could see the incoming ICBM. To watch this critical event happen in our simulation, I placed a sensor on the kill vehicle. As the EKV traversed it trajectory and headed on a collision course with the ICBM, the sensor's continued lock on the target was visualized in *STK* much like the track radar displayed it had a lock on the ICBM. A white line connected the two objects, and a while square was seen around the locked-on vehicle.

I was not able to generate an approximate model of the EKV launching without the outputted *Simulink* data as I had done with the IR sats and the radars. Neither *STK*'s ballistic propagator, nor *MFT*'s ballistic propagator was precise enough to model an intercept missile. Therefore, a visual how the EKV attacks the ICBM can not be generated until the EKV trajectory data is output from *Simulink* in a text file.

Modeling of GPS sats and battle manager

The GPS satellite model was the simplest model, aside from the battle manager, to integrate within *STK*. All I required to model the GPS satellites was each satellite's position and

velocity at each time step. Surachai Sukchoo supplied this information to me from a *MATLAB* module in a text file. Since the GPS satellites only pass information to the EKV, only the satellites themselves need to be visualized.

Originally, I planned to model two EKVs and their positions; one based the *Simulink* output and the other based on the GPS truth data, that is, where the GPS satellites thought the EKV was. However, since Surachai Sukchoo's GPS model was so accurate, I felt it unnecessary to display both trajectories.

Like the GPS satellite model, the battle manager model did not require any elaborate displays in *STK*. Therefore, I simply placed a facility in Colorado Springs, USA that represented the battle manager's location.

Analysis and conclusions

Troubles encountered during mission

Once *MFT* was installed properly, and once I learned all the nuances of the *STK/MFT* interface, the addition of models was fairly easy. Further, once all the pieces to our simulation puzzle were put together, *STK*'s promise of making analysis of complex scenarios came true. This is not to say, however, that my experience with this software was a walk-in-the-park. Below, I provide a quick summary of the major problems I encountered.

Had it not been for these problems, I would have had more time to add fidelity to visualization portion of our GBI simulation.

1. The *Missile Flight Tool* module is not provided on the standard *Satellite Tool Kit* CD; *MFT* requires its own separate CD.

2. To access any *STK* module, such as *Missile Flight Tool*, one must have a password and either an expiration date or a Host ID number for his/her computer station.

2. Most likely, one will have to manually install the password and Host ID number/expiration date. This requires the user to be either extremely familiar with his/her computer or extremely friendly with the Analytical Graphic's technical support staff!

4. Any changes a user wishes to make to his/her *MFT* scenario after he/she has exported the scenario to *STK* requires that the user begin the simulation completely anew.

Growth possibilities

Given more time to work on this simulation, there a number of other scenarios we could model and simulate. For example, it is possible to model the random affects of wind magnitudes and directions on the ICBM and EKV as they travel along their trajectories. Moreover, it is possible to include several different launch sites into our simulation. Obviously, though, this would require the addition of more search and track radars and more IR satellites.

REFERENCES

[1] *Missile Flight Tool User's Manual and Tutorial*, Version 2.0, Applied Technology Group, Alabama, 1997, pp. 1-1, 2-8.

[2] "USAF Fact Sheet 96-09," http://www.af.mil/news/factsheets/LGM 30 Minuteman III.html, April 1996.

[3] "USAF Fact Sheet 96-08," http://www.af.mil/news/factsheets/LGM 118A Peacekeeper.html, April 1996.

[4] Marshall, S., and Patrick, R., *STK User's Manual*, Analytical Graphics, Pennsylvania, 1997, pp C-9 - C-17.